L6: Entry 10 of 11 File: USPT Nov 5, 1996

DOCUMENT-IDENTIFIER: US 5572634 A
TITLE: Method and apparatus for spatial simulation acceleration

Brief Summary Text (40):
FIG. 3 is the Collision Detection Method 300 for solving the collision detection problem for one simulation step within a spatial simulation. In the Object Processing Step 500, object's bounding boxes are compared to each other to determine which objects may intersect. This step 500 generates a list of sets of possibly intersecting objects. Then 404, 406, each set of possibly intersecting objects is processed. In the Geometry Processing Step 900, the geometry representing the surfaces or volumes of objects within a set is processed. This step 900 finds pairs of geometry primitives (in different objects) that have intersecting bounding boxes. Potentially, such primitive pairs may intersect. Then 408, 410, each primitive pair is processed. In the Analytic Geometry Step 412, the specific geometric data of the geometry primitives are used to determine if the primitives actually intersect. If 414 the primitives actually intersect, then a collision has been found 416.

Brief Summary Text (41):
This document describes a method and apparatus for accelerating the Object Processing Step 500 and the Geometry Processing Step 900. This is done with a new type of MCCAM, called a Geometry CAM 1000 (hereinafter abbreviated GCAM 1000). The GCAM 1000 is a multiplicity of GCAM Words 1100, each GCAM Word 1100 including a multiplicity of Coordinate Fields 1200 that perform arithmetic comparisons. The words within the MCCAM of the Duluk Patent also have multiple fields that perform arithmetic comparisons. However, the GCAM 1000 additionally includes mechanisms for groups of GCAM Words 1100 to function together in storing a geometry primitive. Also, the GCAM 1000 includes a mechanism allowing geometry primitives to share vertices. Sharing vertices is important because a commonly used type of spatial primitive, the polygon mesh, shares vertices between polygons. The GCAM 1000 is used to find collisions between geometry primitives (or objects) by detecting intersections between bounding boxes around the geometry primitives (or objects).

Detailed Description Text (5):
The Object Processing Step 500 makes bounding volumes around each object, and tests these bounding volumes for intersection. If the bounding volumes for two objects intersect, then the actual geometry of the objects may intersect. As a new bounding volume is processed, its position in space is compared to the position of all the other bounding volumes. These comparisons are done in parallel by the GCAM 1000, and bounding boxes (defined later) are used as an approximation for the bounding volumes. The output of the Object Processing Step 500 is a list of sets of leaf objects whose bounding boxes intersect.

Detailed Description Text (7):
The Geometry Processing Step 900 takes the list of sets of objects output by the Object Processing Step 500 and processes the geometry primitives within the leaf objects. Bounding volumes around geometry primitives are generated, and then compared to each other to determine intersection. Just as in the Object Processing Step 500, the comparisons are done in parallel by the GCAM 1000 which uses bounding boxes (defined later) as an approximation for the bounding volumes. The output of the Geometry Processing Step 900 is pairs of geometry primitives whose bounding boxes intersect.

Detailed Description Text (15):
For two bounding boxes 406, 408 to intersect, their extents along each axis must overlap. In FIG. 4, the x extents overlap, but the y extents do not. Therefore, these two bounding boxes 406, 408 do not intersect. Thus, the condition of overlapping object

bounding boxes, denoted by the logic value ObjectBoxHit, is the conjunction (i.e., the "and") of overlapping of the three extents. This is shown as Equation 1. ##EQU1##

Detailed Description Text (19):
The logic inverse of ObjectBoxHit(Strd,New) is ObjectBoxMiss(Strd,New), which indicates the bounding boxes do not overlap. ObjectBoxMiss is generated by Equation 5, a disjunction (i.e., an "or") of six terms. ##EQU5##

Detailed Description Text (20):
FIG. 5 is a method flow diagram for the Object Processing Step 500 which compares a set of object bounding boxes. At the beginning of a simulation step, the GCAM 1000 is initialized 502, which wipes it clean of all contents. Then, each object is processed by declaring it a "new" object 504, making a bounding box around it 506, subjecting the bounding box to the GCAM Object Intersection Test 508, and then storing 510 the bounding box into the GCAM 1000. The operation of getting 504 a new object may include translating it from its object coordinate system to the world coordinate system. The GCAM Object Intersection Test 501 compares the new object's bounding box to all previously stored bounding boxes. This 501 is done by getting 512 each previously stored bounding box, testing for intersection 514 using Equation 3, Equation 4, Equation 5, or an equivalent equation, and labeling 516 (or adding to a list) objects that may actually intersect because their bounding boxes intersect.

Detailed Description Text (22):
Without parallelism, the method flow method of FIG. 5 would take N(N-1)/2 bounding box comparisons for N objects, which is O(N.sup.2) operations (O(N.sup.2) is read "order N.sup.2 "). However, the GCAM 1000 is a set of parallel hardware comparators, and reduces the GCAM Object Intersection Test 508 to a single operation. This changes the method flow method of FIG. 5 to O(N).

Detailed Description Text (23):
As a possible means for further acceleration, only the bounding boxes of moving objects are input to the GCAM 1000, while the bounding boxes of non-moving objects are stored unchanged in the GCAM 1000. This can be done because non-moving objects cannot have new collisions with each other. If this is done, initialization 502 of the GCAM 1000 deletes the bounding boxes of only the moving objects, leaving the bounding boxes of non-moving objects stored in the GCAM 1000. Then, only the moving objects are used as new objects 504 for the GCAM Object Intersection Test 501. This acceleration can be done if the GCAM 1000 is not fully used for collision detection testing on finer geometry primitives for pair of objects whose bounding boxes intersect.

Detailed Description Text (50):
The GCAM apparatus must be able to store and compare both bounding boxes and polygon meshes. Bounding boxes can be thought of as a two-point polygons whose vertices are opposite corners of the bounding box. Hence, as bounding boxes are input to the GCAM 1000, they can be properly processed if Equation 11 is modified to ignore C(n, m-2). This can be done by supplying an additional signal for disabling this one set of Comparison Result Bits 1104. However, a more general solution allows any of the Comparison Result Bits 1104 to be disabled. ##EQU12##

Detailed Description Text (55):
An alternate solution to Equation 14, shown in Equation 15, is to add a single disable signal, which disables only the third vertex. Thus, this disable bit, called DisableOldest(n) 1508, is used to chose between a polygon with three vertices (a triangle) and a polygon with two vertices (a bounding box). The GCAM apparatus described hereinafter includes the solution shown in Equation 15, however, that of Equation 14 can be easily substituted.

Detailed Description Text (65):
The Comparison Result Bits 1104 are input to their respective Coordinate Miss Detect 1300, which then determines if that respective coordinate causes the stored item (bounding box, polygon, etc.) to miss (i.e., not collide with) the newly input item. The miss condition is output from each Coordinate Miss Detect 1300 on its corresponding output signal XPlyMs(n) 1132, YPlyMs(n) 1134, or ZPlyMs(n) 1136. These outputs are dependent on Comparison Conjunctions from previous GCAM Words 1100, and these values are input on the busses XVrtxMsOut(n-1) 1142, YVrtxMsOut(n-1) 1144, and ZVrtxMsOut(n-1) 1146, which come from the previous GCAM Word 1100. Hence, the GCAM Word 1100 outputs its Comparison Conjunction values on XVrtxMsOut(n) 1152, YVrtxMsOut(n) 1154, and ZVrtxMsOut(n) 1156.

Detailed Description Text (77):
If the extent of the stored polygon (x, y, or z extent) is wholly less than (does not overlap) the extent of the new polygon (same coordinate axis), then the signal CordMsDueToL(n) 1340 is asserted. If the stored extent is wholly greater than the new one, then CordMsDueToG(n) 1342 is asserted. If either of these conditions are true, then the stored polygon (or bounding box, etc.) cannot intersect, and CordPlyMs(n) 1132, 1134, or 1136 is asserted.

Detailed Description Text (81):
FIG. 14 is a circuit for the Polygon Miss Detect 1400 portion of the GCAM Word 1100. The outputs 1132, 1134, 1136 from the Coordinate Miss Detects 1300 are input to the Polygon Miss Detect 1400, which computes the value of PlyBoxMs(n) 1402, as defined in Equation 12, Equation 14, or Equation 15. This computation is done as the disjunction of five signals. Included in the disjunction is NewPlyInvld 1404 which indicates that the newly input vertex does not correspond to a valid polygon (or bounding box, etc.).

Detailed Description Text (83):
FIG. 15 is a circuit for the Word Status Flags 1500, a portion of the GCAM Word 1100 that stores the word's 1100 status bits. The status bits include: InvalidData(n) 1502, StrdPlyInvld(n) 1504, ReplaceOldest(n) 1506, and DisableOldest(n) 1508. InvalidData(n) 1502, when asserted, indicates the data stored in the word 1100 is invalid, and can be overwritten. StrdPlyInvld(n) 1504, when asserted, indicates the vertex stored in the word 1100 does not correspond to a valid polygon (or bounding box, etc.). ReplaceOldest(n) 1506, when asserted, indicates that the vertex stored in the word 1100 has replaced the oldest vertex of the previous polygon in the mesh. DisableOldest(n) 1508, when asserted, indicates that the completed by the word 1100 does not use the oldest vertex of the previous polygon in the mesh (this is generally used for bounding boxes).

Detailed Description Text (90):
When a read operation is performed, SelFirstInvalidDataWrd 1616 is deasserted, causing the multiplexor 1622 to select the Hit(n) 1602 flag as the value for SelReq(n) 1502. The Priority Resolver 1800 finds the first GCAM Word 1100 where SelReq(n) 1502 is asserted, and asserts that word's 1100 WrdSel(n) 1704, thereby making that word's 1100 stored polygon (or bounding box, etc.) selected for reading, leaving all other word's WrdSel(n) 1704 deasserted. Since the selected item can span many words 1100, each of those words 1100 must be chosen in turn for the read operation. The choosing is done by asserting one of three control signals: 1) ReadSelectedOldest 1611, which selects the oldest vertex in the polygon; 2) ReadSelectedMiddle 1612, which selects the middle vertex in the polygon; or 3) ReadSelectedWrd 1613, which selects the newest vertex in the polygon, namely the one with WrdSel(n) 1704 asserted. This causes one GCAM Word 1100 to assert its WrdReadEn(n) 1204 by the and-or circuit 1624 that performs Equation 15. ##EQU16## Hence, for the read operation, one of the following words 1100 are generally read: 1) the nth word 1100 where WrdSel(n) 1704 asserted, containing the newest vertex; 2) the (n-1)th word 1100, one prior to the word 1100 where WrdSel(n) 1704 asserted, containing the middle vertex; or 3) the (n-2)th word 1100, second prior to the word 1100 where WrdSel(n) 1704 asserted, containing the oldest vertex. The exception to this is in determining the oldest vertex, which maybe stored in a word 1100 prior to the (n-2)th. Since ReplaceOldest(n) 1508 maybe deasserted, then the oldest vertex from the previous polygon in the mesh is kept, and the middle vertex is replaced. If this is the case, then the (n-3)th word 1100 is read rather than the (n-2)th. However, if ReplaceOldest(n-1) 1508, in the previous word 1100, is also deasserted, then the (n-4)th word 1100 is read. Saving the oldest vertex from the previous polygon propagates back to prior words 1100 as long as ReplaceOldest(n) 1508 is deasserted in those words. The need to do this propagation is indicated by the assertion of PropagateOldest(n) 1634, which tells the previous word 1100 to turn off its output, ReplaceOldest(n-2) 1508.

Detailed Description Text (92):
While the last word 1100 for a polygon (or bounding box, etc.) is read, Hit(n) 1602 is cleared in the selected GCAM Word 1100 by asserting ClearSelectedWrdHit 1615. This allows subsequent reads to select another polygon.

Detailed Description Text (107):
In addition to the three spatial dimensions of x, y, and z, a fourth dimension can be added: time. The time coordinate for a vertex would be stored in a fourth Coordinate Field 1200, and all the concepts previously discussed herein are expanded to four dimensions. Adding time as a fourth dimension allows the simulation time steps for objects to be independent. For example, an object could follow a piecewise linear

trajectory through space, where each linear piece would correspond to a four-dimensional bounding box. Then, as the set of bounding boxes for all objects is processed, detected object collisions would be bounded by a time interval within the simulation.

CLAIMS:

10. A polygon collision miss detector structure comprising:

means for receiving from said plurality of coordinate miss-detect means a plurality of coordinate polygon miss detect signals; and

means simultaneously computing the disjunctive logical OR of said plurality of coordinate polygon miss detect signals and generating a polygon box mis-detect signal when any one of said plurality of coordinate miss-detect signals indicate that the coordinate extent of said stored polygon does not overlap the coordinate extent of said new polygon.

13. A word control logic structure comprising:

register means for storing a value derived from said polygon box miss signal output from said polygon miss-detect means of a particular word in response to a store hits/misses control signal generated during a clock cycle when a spatial query is performed, a value being stored in each said GCAM word simultaneously;

means responsive to said priority resolver means for selecting a single word for writing to and enabling writing of data to predetermined fields (coordinate and tag) of said selected word; and

means responsive to said priority resolver means and a vertex read select control signal for selecting a single word for reading from and enabling reading of data from predetermined fields of said selected single word.

19. A geometrical content addressable memory (GCAM) logic circuit comprising:

a plurality of word logic means for storing vertex coordinate values related to a geometry item and for computing geometry item coordinate collision results;

control means for controlling movement of data values into said word logic means and for controlling output of object collision result data from said GCAM; and

means for receiving x-, y-, and z-coordinate values for each geometry item from an external coordinate source;

each said word logic means including:

x-coordinate field storage and comparison logic means including register means for storing an x-coordinate and arithmetic comparison means for arithmetically comparing a new x-coordinate value present on an x-coordinate bus with said x-coordinate value stored in said register means and to determine if said stored value is greater-than or less-than said bus x-coordinate, and for generating an x-coordinate comparison result based on said comparison;

y-coordinate field storage and comparison logic means including register means for storing a y-coordinate and arithmetic comparison means for arithmetically comparing a new y-coordinate value present on a y-coordinate bus with said y-coordinate value stored in said register means and to determine if said stored value is greater-than or less-than said bus y-coordinate, and for generating a y-coordinate comparison result based on said comparison;

z-coordinate field storage and comparison logic means including register means for storing a z-coordinate and arithmetic comparison means for arithmetically comparing a new z-coordinate value present on a z-coordinate bus with said z-coordinate value stored in said register means and to determine if said stored value is greater-than or less-than said bus z-coordinate, and for generating a z-coordinate comparison result based on said comparison;

an x-coordinate collision miss detect field logic means associated with said x-coordinate field storage and comparison logic means for receiving said x-coordinate